**Department of Electrical Engineering**

|  |  |
| --- | --- |
| **Faculty Member:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Dated: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
|  |  |
| **Course/Section:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Semester: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |
|  |  |

**EE-330 Digital Signal Processing**

**Lab#9 FIR filter design using Windowing**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | **PLO4-CLO4** | | **PLO5-CLO5** | **PLO8-CLO6** | **PLO9-CLO7** |
| **Name** | **Reg. No** | **Viva / Quiz / Lab Performance** | **Analysis of data in Lab Report** | **Modern Tool Usage** | **Ethics and Safety** | **Individual and Team Work** |
|  |  | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** | **5 Marks** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**Lab9: FIR filter design using Windowing**

**Objectives**

* The principal objective of this lab is to demonstrate FIR Filter design using windowing.

In addition, linear phase filters will also be demonstrated:

**Lab Instructions**

* The students should perform and demonstrate each lab task separately for step-wise evaluation (please ensure that course instructor/lab engineer has signed each step after ascertaining its functional verification)
* Each group shall submit one lab report on LMS within 6 days after lab is conducted. Lab report submitted via email will not be graded.

. Students are however encouraged to practice on their own in spare time for enhancing their

**Lab Report Instructions**

All questions should be answered precisely to get maximum credit. Lab report must ensure following items:

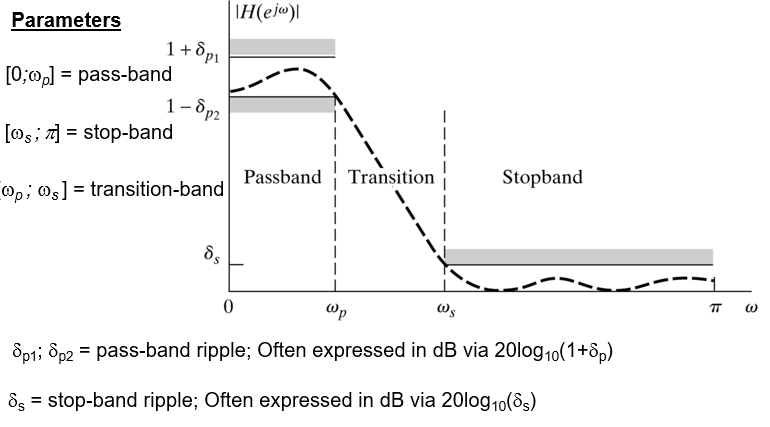
Lab objectives

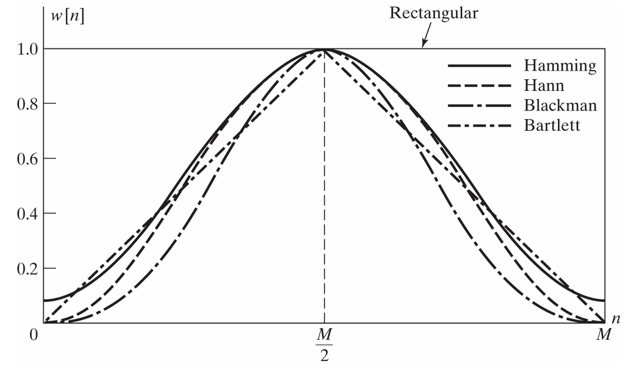
MATLAB/C codes

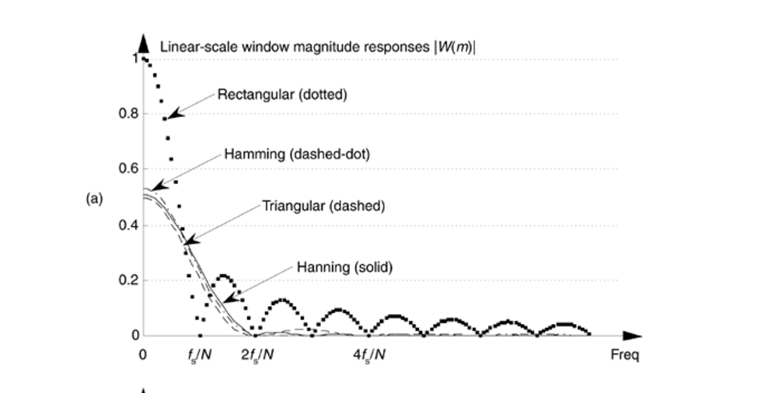
Results (graphs/tables) duly commented and discussed Conclusion

# FIR Window Filter Design with MATLAB

## 







## LINEAR-PHASE FIR FILTER:

Shapes of impulse and frequency responses and locations of system function zeros of linear-phase FIR filters will be discussed. Let ,  be the impulse response of length (or duration) *M*. Then the system function is



which has  poles at the origin *z = 0* (trivial poles) and  zeros located anywhere in the z-plane. The frequency response function is



**WINDOW DESIGN TECHNIQUES**

Summary of commonly used window function characteristics

|  |  |  |  |
| --- | --- | --- | --- |
| Window  Name | Transition Approximate | Width  Exact Values | Min. Stopband Attenuation |
| Rectangular |  |  | 21 dB |
| Bartlett |  |  | 25 dB |
| Hanning |  |  | 44 dB |
| Hamming |  |  | 53 dB |
| Blackman |  |  | 74 dB |

MATLAB provides several routines to implement window functions discussed above table.

* w = boxcar(n) returns the n-point rectangular window in array w.
* w = triang(n) returns the n-point Bartlett (Triangular) window in array w.
* w = hanning(n) returns the n-point symmetric Hanning window in a column vectorin array w .
* w = hamming(n) returns the n-point symmetric Hamming window in a column vector in array w.
* w = blackman(n) returns the n-point symmetric Blackman window in a column vector in array w.
* w = kaiser(n,beta) returns the beta-valued n-point Kaiser window in array w.

Using these routines, we can use MATLAB to design FIR filters based on the window technique, which also requires an ideal lowpass impulse response  as shown below.

function hd=ideal\_lp(wc,M);

% Ideal LowPass filter computation

% --------------------------------

% [hd] = ideal\_lp(wc,M);

% hd = ideal impulse response between 0 to M-1

% wc = cutoff frequency in radians

% M = length of the ideal filter

%

alpha = (M-1)/2;

n = [0:1:(M-1)];

m = n - alpha +eps; % add smallest number to avoid divided by zero

hd = sin(wc\*m)./(pi\*m);

To display the frequency domain plots of digital filters, MATLAB provides the **freqz** routine. Using this routing, we can developed a modified version, called freqz\_m, which returns the magnitude response in absolute as well as dB scale, the phase response, and the group delay response as shown below.

function [db,mag,pha,grd,w] = freqz\_m(b,a)

% Modified version of freqz subroutine

% ------------------------------------

% [db,mag,pha,grd,w] = freqz\_m(b,a)

% db = relative magnitude in dB computed over 0 to pi radians

% mag = absolute magnitude computed over 0 to pi radians

% pha = Phase response in radians over 0 to pi radians

% grd = Group delay over 0 to pi radians

% w = 501 frequency samples between 0 to pi radians

% b = numerator polynomial of H(z) (for FIR: b=h)

% a = denominator polynomial of H(z) (for FIR: a=[1])

%

[H,w] = freqz(b,a,1000,'whole') ;

H = (H(1:1:501))'; w = (w(1:1:501))';

mag = abs(H);

db = 20\*log10((mag+eps)/max(mag));

pha = angle(H);

grd = grpdelay(b,a,w);

### Lab Task 1:

Design the following digital bandpass filter.



These quantities are shown in the following figure.



Follow the steps

* Find transition width
* Calculate M
* Create ideal band pass filter with two ideal low pass filter(for ideal lowpass use hd=ideal\_lp(wc,M) code given Above )
* Implement appropriate window (use the table )
* Multiply window coefficient with your ideal band pass filter
* Plot the ideal impulse response
* Plot actual impulse response
* Plot frequency response

MATLAB will help do most of the work with its “fir1” function built into the Signal Processing Toolbox. For an N-length FIR filter the general form of the fir1 function is:

*b = fir1(N-1, Wn, filter\_type, window);*

*b≡ filter numerator coefficients*

*Wn≡ normalized cutoff frequency (or frequencies) of the filter*

*filter\_type≡lowpass, highpass, bandpass, bandstop*

*window≡ the window coefficients, e.g. Hamming, Blackman, etc.*

Lowpass and highpass filters have only one cutoff frequency, fc. Thus, Wn needs to be a scalar (1 x 1 matrix) if you are designing a lowpass or highpass filter. For bandpass and bandstop you need to specify two cutoff frequencies and Wn will need to be a two-element row vector. If Wn specifies one cutoff frequency then the default filter type will be a lowpass filter. To make a high-pass filter you need to specify the filter\_type as ‘high’. If Wn specifies two cutoff frequencies then the default filter type will be a bandpass filter. To make a bandstop filter you need to specify the filter\_type as ‘stop’. The default window is a Hamming window. Different window types that you can specify include boxcar (rectangular), blackman, bartlett, hanning, kaiser, and more.

Here are some examples of using the fir1 command:

*b = fir1(N-1, 0.5); ‘lowpass filter using Hamming window*

*b = fir1(N-1, 0.5, blackman(N)); ‘lowpass filter using Blackman window*

*b = fir1(N-1, 0.5, ‘high’, boxcar(N)); ‘highpass filter using rectangular window*

*b = fir1(N-1, [0.2 0.5]); ‘bandpass filter using Hamming window*

*b = fir1(N-1, [0.2 0.5], ‘stop’, hanning(N)); ‘bandstop filter using Hanning window*

One very important detail to note is that the MATLAB wants the cutoff frequencies normalized to half the sampling frequency, i.e. Fs/2. Wn should NOT be normalized to Fs! (Keep in mind though that when estimating how many coefficients are needed for a filter we are using a transition width that is normalized to Fs.)

## Lab task 2

Design a lowpass FIR filter with a passband cutoff frequency of 1 kHz, a stopband edge at 4.3 kHz, and a sampling frequency of 10 kHz. We will use a Hamming window. The transition width is the difference of the stopband edge and the passband edge. Thus, the normalized transition width Δf is the number of coefficients can then be estimated as

We must now determine the normalized cutoff frequency for MATLAB. Remember this is to be normalized to Fs/2. Here our cutoff frequency is 1 kHz .

### Lab Task 2:

Design Bandpass Filter with following specification using fir1 function.

*Fs = 48 kHz*

*Passband Cutoff Frequencies = 8 kHz & 16 kHz*

*Stopband Edge Frequencies = 7 kHz & 17 kHz*

*Hamming Window*